A Toolkit for Specifying Service Level Agreements for IoT Applications

Awatif Alqahtani[1], Pankesh Patel[2], Ellis Solaiman[1], Rajiv Ranjan[1]
[1]**School of Computing Science, Newcastle University, Newcastle, UK**
[2]**Fraunhofer CESE, College Park, Maryland, USA**
{a.alqahtani, ellis.solaiman, raj.ranjan}@ncl.ac.uk; ppatel@cese.fraunhofer.org

DEMO ABSTRACT

Service level agreement (SLA) in IoT involve many providers within the same architectural layer or between different architectural layers. For example, sensors that collect a patient's data are supplied by an actor that is different from the actor that provides a networking service in order to feed data to an IoT application, which in turn is provided by another actor. An actor is any participant in delivering an IoT application/service: it might be software, hardware or a human being. Each actor has responsibilities, abilities and requirements. To create an SLA, these different actors need to be considered, ensuring that each actor involved is delivering the required job within a certain level of Quality of Service (QoS), as well as receiving its requests within its QoS constraints. Therefore, this kind of dependability needs to be reflected and captured, and currently available SLA specification languages do not reflect this need. Several key challenges need to be considered to enable a shift from previous SLA specification languages to an SLA specification language for IoT applications. One of the challenges is the multi-layer nature of IoT Applications. Therefore, in this demo, we demonstrate a toolkit for creating SLA specifications for IoT applications. The toolkit is used to simplify the process of capturing the requirements of IoT applications. We present a demonstration of the toolkit using a Remote Health Monitoring Service (RHMS) use-case. The toolkit supports the following: (1) specifying the Service-Level Objectives (SLO) of an IoT application at the application level; (2) specifying the workflow activities of the IoT application; (3) mapping each activity to the required software and hardware resources and specifying the constraints of SLOs and other configuration- related metrics of the required hardware and software; and (4) creating the composed SLA in JSON format.

KEYWORDS - Clouds and Edge Computing and Applications; Service Level Agreement; SLA Specification; IoT.

OUTLINE

1. System Overview

In this section, we present the design goals and the architecture of the toolkit.

1.1 Design Goals

SLA creation is an important and critical step considering the fact that SLA-based service discovery, negotiation, monitoring, management and resource allocation rely on what has been specified within the SLA. As a result, we have developed a toolkit that enables service consumers to specify their QoS requirements and express them as service level objectives (SLOs), as well as specifying some configuration- related metrics for each software/hardware component of the system. We have considered the following features as design goals of the toolkit:

- Expressiveness: We aim to provide a rich list of domain specific vocabularies to allow fine-grained SLA specification.
- Generality: We aim to consider common components or layers of IoT architecture (IoT, Edge and Cloud).
- Extendibility: The tool is to some extent extendable, because it has been designed to allow anyone who is interested in customising/enhancing the SLA according to his/her application-specific need to add or delete

activity/metrics without changing the programming code. It is possible to add/delete/change activity/metrics using an attached Excel file, and these changes can be reflected dynamically. The Excel file preserves the schema of SLA components (e.g., workflow activities and their related software and hardware requirements).

- Simplicity: Providing a GUI enables users to specify their requirements without needing prior knowledge of a machine-readable language such as JSON or XML. Furthermore, the tool allows users to specify an SLA in the same data-flow as their application, by allowing users to specify the workflow activity of their application first and then specify the requirements in the same flow of occurrences as the selected activities.

1.2 System Architecture

The abstracted design and architecture of the toolkit is depicted in Figure 1. The overall architecture comprises three basic layers:

- GUI Layer, which includes the user interface components. It displays user interface components as a sequence of forms that guide the user through well- defined steps.
- Programming Layer, which encapsulates the programming modules in order to serve the GUI layer by providing the required functionalities.
- Data Layer, which encapsulates the required data as an input to the tool or output of the tool. It includes:
  - An Excel file as an input, which provides data that describes the SLO and configuration metrics related to the software and hardware requirements of each activity. The Excel file has new vocabularies which provide fine-grained details regarding the associated software and hardware components of each workflow activity.
  - A JSON document as an output, which represents the SLA document. The specification is related to the SLOs at the application level followed by a specification related to each activity, which includes the SLOs and configuration related metrics required for the programming model (e.g., stream processing) and deployment layer (e.g., Cloud layer).

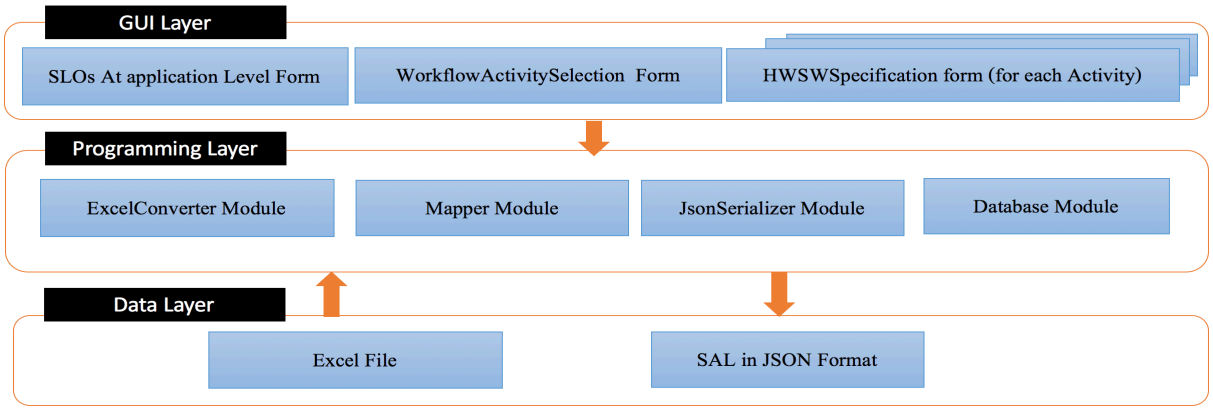

Fig. 1 Layered Architecture

2. Demonstration

The goal of our demonstration is to show how to create an SLA for an IoT application using a GUI-based toolkit. The tool considers domain-specific metrics for SLA specification purposes.

2.1 Remote Health Monitoring Service (RHMS) Use-case

To illustrate the usefulness of the toolkit, consider a Remote Healthcare Monitoring Service (RHMS) where patients wear sensors and accelerometers to measure their heart rate and sugar levels (capture data with high accuracy guarantees), reminding them of the time to take medications and detecting abnormal activity such as falling down (analyse incoming data on fly under low latency constraints). Data transferring from sensors to the Cloud layer requires 100% network connectivity. Patients can register in RHMS and pay for the service to monitor their health remotely and alert their carers and doctors if their health is in a critical condition. Subscribed patients are looking for a service that can satisfy the following high-level requirement: detecting abnormal activity (such as falling down) within x milliseconds; ambulance, carers and doctors to be contacted within y minutes [1]. Adherence to SLA's

constraints of RHMS is a critical process. For example, if there was a delay in the network, it would lead to a late response at the front-end which exceeds what the consumer was expecting. From the above scenario, it can be seen that to achieve the high-level requirement, many nested-dependent QoSs should be considered. Therefore, the toolkit defines a multilayer specification with new vocabulary to allow for specifying the constraints for all back-end services, which cooperate to deliver the front-end service.

2.2 Demonstration Scenario

Our demonstration would involve the following steps:

- Specify the SLOs of the Application at the application level: The tool will display a predefined list of possible SLOs as a check list. Users can check the SLOs that they are interested in and specify the priority level (high, low, or normal) as well as the threshold value of the QoS metric of the SLO (Figure 3-a). In RHMS, users can quantify the SLOs by specifying the acceptable threshold value of the related Quality of Service metrics. For ex- ample, the objective of minimizing response time can be specified by selecting the preferred value for each of the following attributes related to the minimizing response time objective: priority (e.g., high); required level (e.g., less than); value (e.g., 60); unit (e.g., seconds).
- Select the workflow activity based on the application scenario: There is a predefined list of activities which are part of many IoT applications' workflow (e.g., capture event of interest; ingest data; analyse large-scale real- time data activity). The tool displays the predefined activity and then the user can select the ones that are included in their application workflow activities and connect them in a way that reflects the data flow of the application. Connecting the activity preserves the dependencies between activities for future work related to performance modelling. For example, the workflow activity of RHMS can consist of the following activities which can be connected, sequentially, in the same order as listed below (Figure 2-b): 1) Capture Event of Interest (EoI). 2) Examine captured EoI. 3) Ingest data activity. 4) Real-time Analysis activity. 5) Store structured data activity. The reason behind considering very standard and common activities is to increase the generality of the tool.
- Specify SLO and configuration metrics related to each of the selected activities: The tool reads the SLO and configuration metrics schema from a predefined Excel file, which has the schema content of what to display for each activity. Then, users can specify the required level/value of an SLO and the configuration metrics for each software/hardware component required to deliver the selected activities (Figure 2-c). In RHMS, for example, when the user selects a "capturing event of interest" activity, the user can then specify the requirements at the IoT devices level, such as sampling rate, battery life and communication mechanism.
- Generate SLA document: Based on what the user has specified in steps 1 and 3, the SLA document will be generated in a JSON format. The generated SLA can be used later on for different purposes, such as service provider discovery, SLA-based monitoring and SLA-based resource allocation. In RHMS, when the user presses finish, a JSON document is generated based on what has been specified, which represents the SLA of the RHMS (Figure 2-d). The JSON document's specification is related to the SLOs at the application level followed by the specification related to each activity, which includes the SLOs and configuration related metrics for the required programming model (such as stream processing) and deployment layer.
- Store generated SLA using a NoSQL database: Since users can choose which metrics to specify by checking/unchecking the metrics, Therefore, generated SLA has different schemas of JSON files being created, due to the heterogeneity of requirements from different users. Therefore, each generated SLA JSON file is stored in a NoSQL database (MonogDB database). The reason behind storing required SLA metrics is for SLA compliance monitoring, which is one part of the SLA's life cycle.

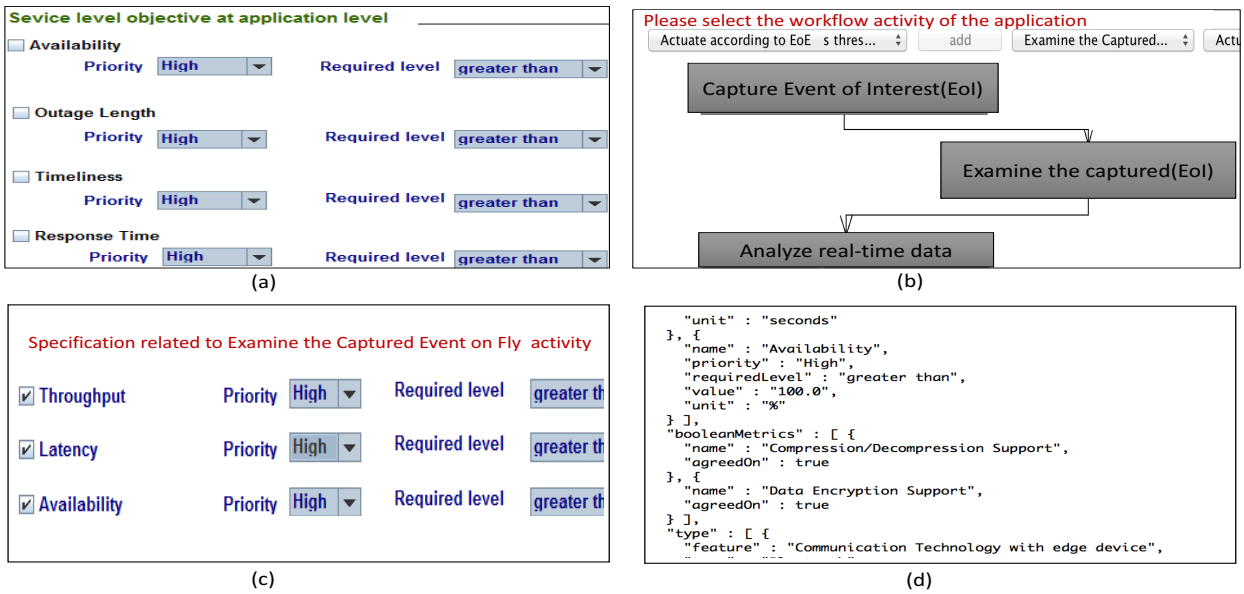

Fig.2 SLA specification toolkit: (a) Specify the SLOs of the Application, (b) Select the workflow activity based on the application scenario requirement, (c) Specify SLOs and configuration metrics related to each of the selected activities, and (d) Generate SLA

References

[1] P. P. Jayaraman, K. Mitra, S. Saguna, T. Shah, D. Georgakopoulos, and R. Ranjan, "Orchestrating quality of service in the cloud of things ecosystem," in *2015 IEEE International Symposium on Nanoelectronic and Information Systems*, Dec 2015, pp. 185–190.

REQUIREMENTS AND TARGET AUDIENCE

Requirement of the Demo:

- JRE (Java Runtime Environment): To run Java JAR file.
- Microsoft Excel: To view the content of the Excel file.

The toolkit is designed to be used by people who are interested in requesting/offering SLA of IoT applications with basic technical knowledge of IoT, Edge and Cloud technologies (e.g., IoT administrators).

DEMO DURATION

The demo will be presented in a 30-minute session (unless otherwise noted).

A/V AND EQUIPEMNT

Laptop and screen